Proceedings of the Fourth International Conference on Machine Learning and Cybernetics, Guangzhou, 18-21 August 2005

TOWARD A SEMANTIC SEARCH ENGINE BASED ON ONTOLOGIES

WEI-DONG FANG, LING ZHANG, YAN-XUAN WANG, SHOU-BIN DONG

Network Engineering and Research Center, South China University of Technology, Guangzhou 510640, China
E-MAIL: {fangwd, ling, yxwang, sbdong}@scut.edu.cn

Abstract:

Semantic search requires a search engine to properly interpret the meaning of a user's query and the inherent relations among the terms that a document contains with respect to a specific domain. We present the framework of such a search engine based on domain ontologies. In this framework, a search request, which can be either a keyword list as in traditional search methods or a query in complex form containing various restrictions to the search, is first processed by a query parser which then finds qualified RDF triples in domain ontologies. Web documents relevant to the requested concepts and individuals specified in these triples are then retrieved by a document retriever. And finally, the retrieved documents are ranked according to their relevance to the user's query. An extended term-document matrix is built to reflect the relevance between documents, concepts/individuals, and terms. Such a matrix makes it possible for the search engine to work even in case that there are no available domain ontologies for user requests.

Keywords:

Semantic search; Information Retrieval (IR); Semantic Web, Ontology; Knowledge Representation; OWL-QL

1. Introduction

With the rapid growth of online information, it becomes more urgent than ever for search engines to be capable of searching web documents based on their content, rather than based on keywords as the traditional search tools do. While present techniques such as Latent Semantic Analysis (LSA) [1] have shown some potential in enabling this kind of discovery, they seem not to be able to meet this challenge well. The reason is: these approaches do not care about the meaning of the keywords that a user provides, and they do not care about the semantic relations among the terms that a document contains with respect to a specific domain.

In a typical search process, a user provides one or more keywords for a search engine, and the search engine, based on some algorithms, finds the documents relevant to these keywords. Although what the user needs are documents containing some information characterised by his/her keywords in a domain, documents retrieved by a search engine are completely based on the mathematical relation between their constituent elements (words, phrases and so on) and the user's keywords, regardless what meanings these elements may have.

We argue that to improve the performance of a search engine in terms of precision and recall, the user's keywords must be properly interpreted according to their meanings in a specific domain; and the inherent relations among the terms in a document must also be considered. In this paper, we present our work toward such goals in the development of a semantic search engine, including its framework and preliminary implementation. Existing Information Retrieval (IR) approaches and the maturing of Semantic Web [2] technologies form the foundation of our work. Specifically, we use ontology [3] to represent domain knowledge; we extend the traditional Term-Document Matrix (TDM) [4] to reflect the relevance between Ontologies, Web documents and terms; and we use Web Ontology Language (OWL) [5] as the ontology definition language and OWL Query Language (OWL-QL) [6] the language for deductive query answering on the ontologies.

In our framework, a search request can be either a list of keywords, or a complex query expression created by the user with the help of a query wizard. In either case, the search request will be first analysed by a query parser, where one or more standard OWL-QL queries are generated. These queries are then sent to a reasoner, which will return a set of RDF (Resource Description Framework) [7] triples containing qualified concepts/individuals in domain ontologies. After that, a document retriever finds all documents that are relevant to these concepts/individuals, and a ranker sorts these documents according to their relevance to the search request before they are brought to the user.

This paper is structured as follows. Section 2 analyses the importance of domain knowledge in semantic search, and how it can help in a search process. Section 3 introduces an extended term-document matrix which we use to quantify the relevance between Web documents, ontology members and terms used by Web documents.

0-7803-9091-1/05/$20.00 ©2005 IEEE

Section 4 describes query processing, showing how our model interprets a query, how it ranks the retrieved documents and how it deals with queries when there are no available ontologies. Finally, section 5 shows the results of a prototype system implemented according to the proposed framework, and concludes this paper by pointing out the future search directions toward a full-fledged semantic search engine.

2. Domain knowledge for semantic search

Traditional search engines do not deal with any domain knowledge, so they do not understand the meaning of a user's search request and the inherent relations among the terms that a Web document contains. This severely limits their abilities to do content-based search. Due to the complexity of natural languages, such as synonyms and polysemes, the precision and recall of the search results can hardly be guaranteed. Although present engines are trying to make up this by means of query expansion or LSA [1], the problem is not touched in nature. How to employ domain knowledge to assist in the search process is the key to semantic search.

In our view, three problems must be addressed to use domain knowledge in a search engine, i.e. a knowledge representation model, the language to define that model and the approach to get support from that model.

For the first problem, we adopt ontology as the knowledge representation model. In computer science, an ontology is a hierarchically structured set of terms to describe a domain that can be used as a skeletal foundation for a knowledge base [3]. In an ontology, concepts, their properties and relationships, constraints and rules are described in a formal language which makes ontologies interpretable to a computer program. A domain ontology provides a vocabulary for representing and communicating knowledge in that domain and a set of relationships that hold among the terms in that vocabulary.

For the second problem, we adopt OWL DL [5] to represent domain ontologies. OWL DL is a sublanguage of OWL which can be directly mapped to Description Logics (DL) [8]. DL is a knowledge representation formalism based on a subset of first-order predicate logic, a declarative formalism for the representation and expression of knowledge. It formalises the nature of knowledge representation logically and provides for sound, tractable reasoning methods founded on a firm theoretical and logical basis. While retaining computational completeness (i.e. all conclusions are guaranteed to be computable) and decidability (i.e. all computations can finish in finite time), OWL DL provides facilities to set up knowledge bases, to reason about their content, and to manipulate them. OWL facilitates greater machine interpretability of Web content than that supported by XML, RDF, and RDF Schema (RDF-S) [9] by providing additional vocabulary along with a formal semantics. Figure 2 is an example of a small ontology about animal and a snippet of its corresponding OWL code is shown in List 1.

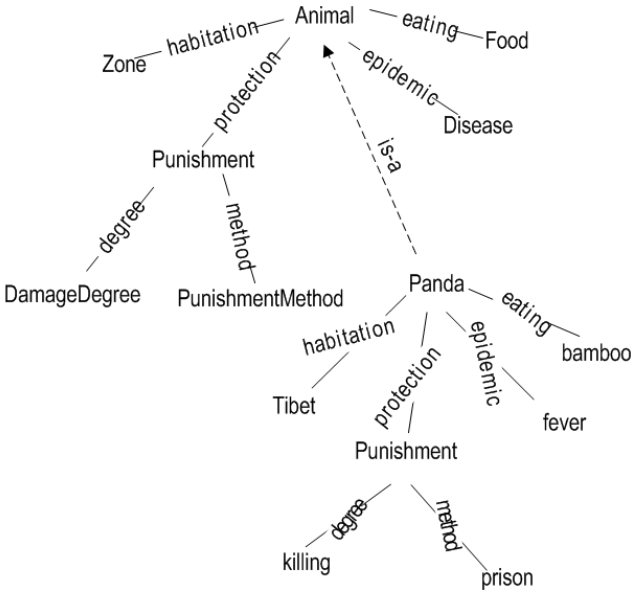

Figure 2. A small ontology about animal

```
<owl:Class rdf:about="#Plant">
   <owl:disjointWith rdf:resource='#Animal'/>
</owl:Class>
<owl:Class rdf:ID="Animal">
   <owl:disjointWith>
      <owl:Class rdf:ID="Plant"/>
   </owl:disjointWith>
</owl:Class>
<owl:Class rdf:ID="Panda">
   <rdfs:subClassOf rdf:resource="#Animal"/>
</owl:Class>
...
<owl:Class rdf:ID="Food"/>
...
```

List 1. A snippet of the OWL code for animal ontology

With DL as the logical foundation, a knowledge base in our framework comprises two components, i.e. a TBox and an ABox (Figure 1). The TBox contains the definitions of all concepts in a domain, while the ABox contains

assertions about named individuals in terms of this vocabulary. Thus a domain ontology includes both concepts and individuals. However, for conciseness, unless specified particularly, we will use *concepts* to denote them both hereinafter.

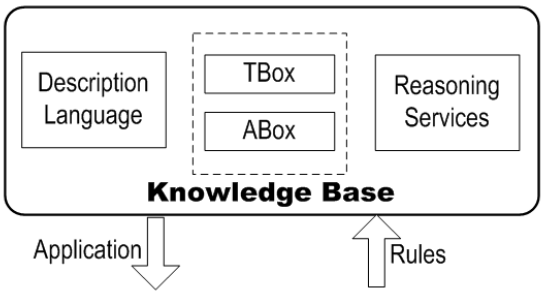

Figure 1. The DL based knowledge base model

For the third problem, because an ontology formally defines the semantic relationships among various concepts in a particular domain in a computer-interpretable language (OWL DL here in our work), these relations can be conveniently employed in a search process. We will show how we specify the relevance between concepts, terms and documents with an extended TDM in section 3 and how we do semantic search by first finding the concepts that satisfy the constraints specified in a user's query and then retrieving the documents which are relevant to these concepts in section 4.

3. The extended Term-Document Matrix

To retrieve Web documents that are related to a user's query and to rank them according to their importance, the relevance between documents and concepts in an ontology must be measured. We quantify such relevance in three steps. First, we represent the relevance between documents and terms with a TDM using the tf-idf weighting scheme; then we define several levels of relevancy between terms and concepts with respect to their positions in an ontology; and finally we get the relevance between documents and concepts by combing the results of these two steps and store it in an extended TDM. These steps are detailed as follows.

Step 1. Calculate the basic TDM

Let N be the total number of documents and n_i the number of documents in which the index term k_i appears. Let $freq_{ij}$ be the raw frequency of term k_i in the document d_j. Then the normalised frequency tf_{ij} of that term in d_j is

$$tf_{ij} = \frac{freq_{ij}}{max_i(freq_{ij})} \tag{1}$$

where the maximum is computed over all terms mentioned in the document d_j. Now let idf_i be the inverse document frequency for k_i given by

$$idf_i = log(\frac{N}{n_i}) \tag{2}$$

The final tf-idf weight of term i to document j is calculated as

$$w_{ij}' = tf_{ij} \times idf_i \tag{3}$$

The TDM is represented as:

$$W' = \{w_{ij}'\} \tag{4}$$

Step 2. Define the levels of relevance between ontology members and documents

We define five relevance levels, including *Direct*, *Strong*, *Normal*, *Weak* and *Irrelevant*. For quantification, each of them can be given a number. For example, in our implemented system, these five levels are given 1.0, 0.7, 0.4, 0.2 and 0.0 respectively. The relevance between an ontology member m (a concept or an individual) and a document d is calculated as

$$R(d,m) = \sum_{i=1}^{N_d} r(t_i, m) \tag{5}$$

where $r(t_i, m)$ is the relevance between m and term i in d, N_d the number of terms in d. The calculation of $r(t_i, m)$ varies subject to the type of m (concept or individual).

(1) For concept

The relevance between a term t and a concept c is:

Direct: if t = c

Weak: if t = parent(c) or t = child (parent(c))

Strong: if t = child(c)

Normal: if t = partner(c)

Irrelevant: for other cases.

Note:

a) parent(c) and child(c) refer to c's parent concept and child concept respectively;

b) partner(c) denotes the filler concept of any c's

roles. For example, in "pandas eat bamboo", bamboo is the partner concept of panda over role eat;

c) for any concept x, t = x means that t is the name of x, or t is the name of one of x's equivalent concepts.

(2) For individual

The relevance between a term t and an individual i is:

Direct: if t = i

Strong: if t = class(i)

Weak: t = instance(class(i)), t != i

Normal: if t = partner(i)

Irrelevant: for other cases.

Note:

a) class(i) refers to the class of instance i;

b) instance(c) refers to the instance of class c;

c) partner(i) refers to the filler instance of any i's roles;

d) for any instance x, t = x means that t is the name of x, or t is the name of one of x's equivalent individuals.

Step 3. Calculate the extended term-document matrix:

$$W = \{w_{ij}\} = \{w'_{ij} \times (1.0 + R_{ij})\} \quad (6)$$

where R_{ij} is the relevance between the i-th term and document j calculated with (5).

The extended term-document matrix reflects the relationship between concepts in a domain ontology, a document corpus, and terms included in the documents. As shown in next section, it makes it convenient to integrate the functionalities of both ontology-based search model and traditional Vector-Space Model [11] in a same engine; and such integration is very meaningful when there are no domain ontologies available for the user requests.

4. Query processing

The search process begins with the parsing of a user's query (Figure 3). If a search request is in form of keyword list, then these keywords would be first treated as concepts in an ongology, and documents that relates to these concepts will be retrieved based on the extended TDM discussed in section 3. Figure 4 illustrates the relations between domain, ontology, concept, keyword, document and user request in a keyword based semantic search. Through a user interface, a user can also submit his/her requests by using a search wizard where he/she is given advanced options for a query. These options may include the ontology server, premises, answer patterns, maximum number of answers, and so on.

In either forms, the request will be parsed into OWL-QL and then sent to the Reasoner which is responsible for query answering.

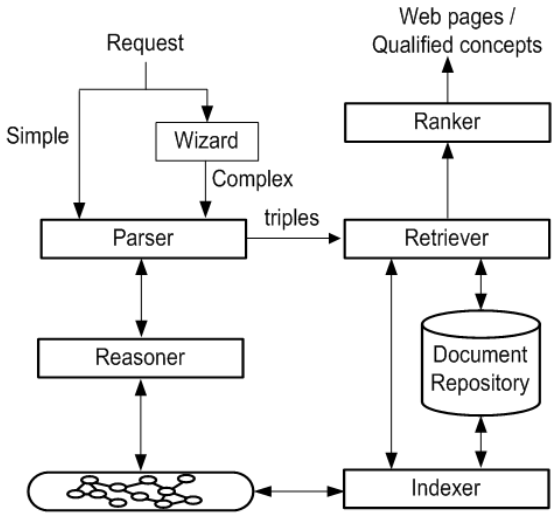

Figure 3. Query processing

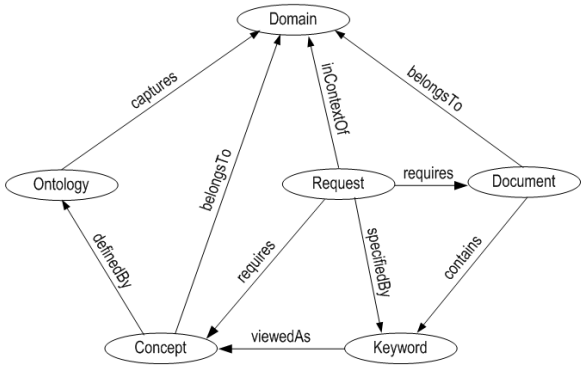

Figure 4. The relations between domain, ontology, concept, keyword, document and user request in a keyword based semantic search

OWL-QL [6] is a formal language. It precisely specifies the semantic relationships among a query, a query answer, and the knowledge base(s) used to produce the

answer. In OWL-QL, a knowledge base K is considered to be a collection of logical sentences K_S that represents a logical theory in which a collection of entailed sentences K_{ES} are true such that $K_S \subseteq K_{ES}$. OWL-QL asks sentences in KES that "satisfy" a given "sentence schema", and to think of using bindings to variables in that sentence schema as specifying answers to the query. List 2 gives a simple query example:

```
Query: ("What pandas in Tibet eat?")
Query Pattern: {(eat ?p ?c) (type ?p Panda) (liveIn ?p Tibet)}
Must-Bind Variables List: (?c)
May-Bind Variables List: ()
Don't-Bind Variables List: ()

Answer Pattern: {(eat "pandas in Tibet" ?c)}
Answer KB Pattern: …
Answer:    ("pandas in Tibet eat bamboo?")
Answer Pattern Instance: {(eat "pandas in Tibet" bamboo)}
Query: …
```

List 2. An query example in OWL-QL

As in the example, the answer to a query contains a set of RDF triples of the form (<property> <subject> <object>), with their order slightly different from the *standard* RDF form (<subject> <property> <object>). Once these triples are returned, a Retriever will get all the documents that correspond to the bound concepts in these triples.

Note that if there is no triple satisfying the query found, the Reasoner will return the original keywords or concept names. For example, if the user searches "what is the food of a kangaroo?", and there is not a "kangaroo" in the ontology, the query engine will return "food, kangaroo".

The ranking algorithm of the retrieved documents combines two factors: the concept weight specified in the user's query and its relevance to a document indicated in an extended TDM as described in Section 3. All concept names (or individual names) returned by the Reasoner (including the original keywords or label names which have no corresponding triples) form a query vector.

A document vector x_i in the extended TDM calculated with (6) is ranked according to the similarity between it and the query vector q. This similarity, sim(q, x_i), is defined as:

$$sim\ (q, x_i) = \frac{q \cdot x_i}{|q| \times |x_i|} \tag{7}$$

From the above descriptions we can see that if there are no ontologies available, our system will work just as a traditional search engine based on VSM. Because in such case, (a) the query parser just returns a keyword list corresponding to what a user submitted in his/her search request; (b) all R(d, m)'s in (5) are zero, thus the extended term-document matrix defined with (6) retrogresses to a normal TDM; and (c) the similarity between a query vector and a document vector is the same as in VSM.

Note that the semantic relationships defined in an ontology are different from the relationships obtained by traditional IR models (such as LSA), where they are calculated mathematically in terms of the constituent characteristics of groups of documents [10]. In these models, any words that satisfy specific parametric conditions are believed to be relevant, no matter what the semantic meanings these words have or what domains they belong to. But the concept relationships defined in a domain ontology are formal and explicit and do not depend on how many documents they are contained or how many times they occur in a document.

To better understand the above difference and the power of ontology-based search, let's take an imaginary example. Suppose John is a famous professor in a university and Mary is his wife. There are many web pages about John stored in the database of a search engine X, but nothing about his wife. Several days ago, Mary donated 1 million dollars by winning a lottery. Then such a message was quickly published on a Web page P and was found by a crawler of X. Now, suppose Kerry wants to get some information about the wealth of John from the Internet, so he tries to search it by X (we assume Kerry is doing this lawfully, e.g. he is a policeman and is just investigating a crime). What will happen? Although X knows Mary has won 1 million, it has no idea about the relationship between John and Mary (because there are not enough documents to calculate the relevance of the couple's names), so it will not give P to Kerry, instead, it will possibly give him 1 million web pages containing the keywords that Kerry provides. However, if X can access the staff ontology of John's university, it can easily learn the relationship between John and Mary, and gives P to Kerry.

In our implemented system, the search engine can not only search the Web for documents which are relevant to user queries, but also display semantic relationships of all involved concepts defined in the ontologies. In addition, if a search request is in form of keyword list, all possible queries in full form (including premise, answer pattern, number restriction, etc.) will be shown to the user along with the search results for the most possible one deduced by the system.

5. Implementation and conclusion

Based on the framework introduced above, we have implemented a prototype system for semantic search.

Compared with traditional search engines, our system can significantly improve the precision and recall of Web searches when necessary domain ontologies are available. Figure 5 shows the comparison between the results of our implemented framework and a keyword-based search system using traditional VSM. The experiment is carried out with a collection of manually classified 3600 documents, covering 3 topics: animal, education and sports and each with 1/3 of the total documents. An animal ontology, which contains 982 concepts, is used in the experiment as the domain knowledge base.

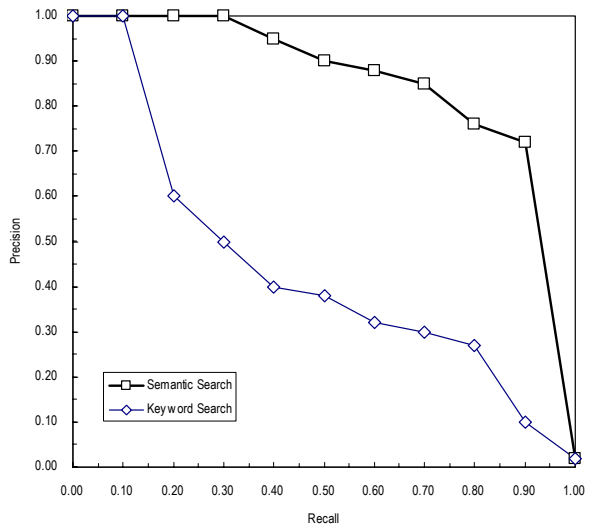

Figure 5. Precision-recall graph of the proposed framework and VSM

The proposed framework can be viewed as an extension of the traditional VSM with semantic support. Future effort around this framework will be focused on the optimisation of the numbers given to the relevance levels (we gave them manually in this paper), and the improvement of the reasoner's performance.

Acknowledgements

The work presented in this paper is supported by the Natural Science Foundation of China under agreement No. 90412015, and the Bioinformatics Grid Project of China Education Department under grant No. CG2003-GA002.

References

[1] Dumais, S. T., Furnas, G. W., Landauer, T. K. and Deerwester, S., "Using latent semantic analysis to improve information retrieval", Proceedings of CHI'88: Conference on Human Factors in Computing, New York, pp 281-285, 1998.

[2] Tim Berners-Lee, James Hendler, Ora Lassila, The Semantic Web, Scientific American, May 2001.

[3] Swartout B., Patil R., Knight K., Russ T, Toward distributed use of large-scale ontologies. Proceedings of the Tenth Knowledge Acquisition for Knowledge-Based Systems Workshop. (KAW '96), Alberta, Canada, November 1996.

[4] William Mill and April Kontostathis, "Analysis of the values in the LSI term-term matrix", Technical report, Ursinus College, 2004.

[5] Sean Bechhofer, Frank van Harmelen, Jim Hendler, et al, OWL Web Ontology Language Reference, http://www.w3.org/TR/2004/REC-owl-ref-20040210/, February 2004,

[6] Fikes, R., Hayes, P., and Horrocks, I. "OWL-QL - A Language for Deductive Query Answering on the Semantic Web", Technical Report Knowledge Systems Laboratory, Stanford University, Stanford, CA, KSL-03-14, 2003.

[7] Resource Description Framework (RDF); World Wide Web Consortium; http://www.w3.org/RDF/, August, 2003.

[8] Franz Baader, Diego Calvanese, Deborah McGuinness, et al., "The Description Logic Handbook", Cambridge University Press, 2003.

[9] "RDF Vocabulary Description Language 1.0: RDF Schema", World Wide Web Consortium, http://www.w3.org/TR/rdf-schema/, October, 2003.

[10] Kiryakov, A., Popov, B., Terziev, et al. "Semantic Annotation, Indexing, and Retrieval", Journal of Web Semantics, Issue 1, Elsevier, pp 47-49, 2004.

[11] M.W. Berry, Z. Drmaˇc, and E.R. Jessup. Matrices, "Vector Spaces, and Information Retrieval". SIAM Review, 41(2), pp 335-362, 1999.